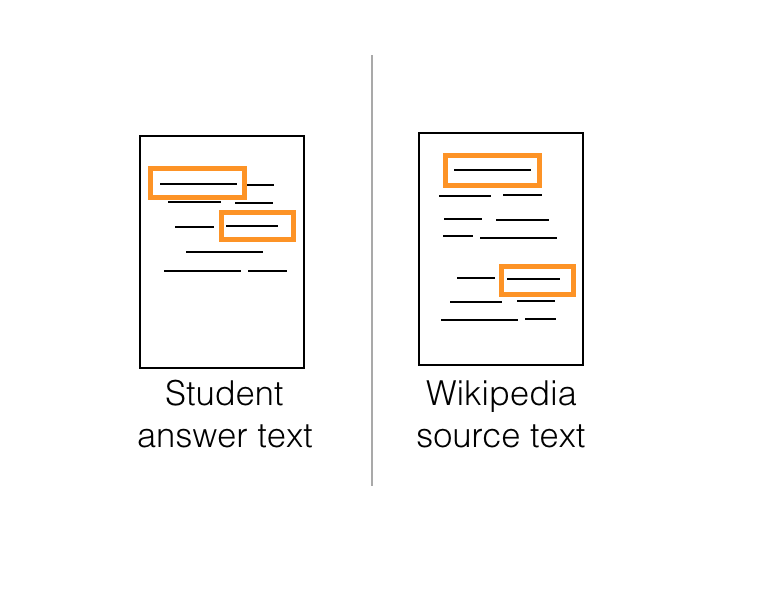
**Plagiarism Detection Project**

In this project, you will be tasked with building a plagiarism detector that examines a text file and performs binary classification; labeling that file as either *plagiarized* or *not*, depending on how similar that text file is to a provided source text. Detecting plagiarism is an active area of research; the task is non-trivial and the differences between paraphrased answers and original work are often not so obvious.

Later in this lesson, you'll find a link to all of the relevant project files.



**Defining Features**

One of the ways you might go about detecting plagiarism, is by computing **similarity features** that measure how similar a given text file is as compared to an original source text. You can develop as many features as you want and are required to define a couple as outlined in [this paper](https://s3.amazonaws.com/video.udacity-data.com/topher/2019/January/5c412841_developing-a-corpus-of-plagiarised-short-answers/developing-a-corpus-of-plagiarised-short-answers.pdf) (which is also linked in the Lesson Resources tab. In this paper, researchers created features called **containment** and **longest common subsequence**.

In the next few sections, which explain how these features are calculated, I'll refer to a submitted text file (the one we want to label as plagiarized or not) as a **Student Answer Text** and an original, wikipedia source file (that we want to compare that answer to) as the **Wikipedia Source Text**.

You'll be defining a few different similarity features to compare the two texts. Once you've extracted relevant features, it will be up to you to explore different classification models and decide on a model that gives you the best performance on a test dataset.

# Containment

One of your first tasks will be to create **containment** features that first look at a whole body of text (and count up the occurrences of words in several text files) and then compare a submitted and source text, relative to the traits of the whole body of text.

## Calculating containment

You can calculate n-gram counts using count vectorization, and then follow the formula for containment:

\frac{{count(\text{n-gram})}\_{A} \cap count (\text{n-gram})\_{S}}{count(\text{n-gram})\_{A}}*count*(n-gram)*A*​*count*(n-gram)*A*​∩*count*(n-gram)*S*​​

If the two texts have no n-grams in common, the containment will be 0, but if all their n-grams intersect then the containment will be 1. Intuitively, you can see how having longer n-gram's in common, might be an indication of cut-and-paste plagiarism.

# Plagiarism Detection

## Project Overview

In this project, you will be tasked with building a plagiarism detector that examines a text file and performs binary classification; labeling that file as either plagiarized or not, depending on how similar the text file is to a provided source text.

This project will be broken down into three main notebooks:

**Notebook 1: Data Exploration**

* Load in the corpus of plagiarism text data.
* Explore the existing data features and the data distribution.
* This first notebook is **not** required in your final project submission.

**Notebook 2: Feature Engineering**

* Clean and pre-process the text data.
* Define features for comparing the similarity of an answer text and a source text, and extract similarity features.
* Select "good" features, by analyzing the correlations between different features.
* Create train/test .csv files that hold the relevant features and class labels for train/test data points.

**Notebook 3: Train and Deploy Your Model in SageMaker**

* Upload your train/test feature data to S3.
* Define a binary classification model and a training script.
* Train your model and deploy it using SageMaker.
* Evaluate your deployed classifier.

## Getting the Project Materials

You have been given the starting notebooks in a Github repository, linked below.

*Since this project uses SageMaker, it is suggested that you create a new SageMaker notebook instance using your*[*AWS console*](https://console.aws.amazon.com/)*and link it to the Github repository*[*https://github.com/udacity/ML\_SageMaker\_Studies*](https://github.com/udacity/ML_SageMaker_Studies)*.*

**The project files are in the Project\_Plagiarism\_Detection directory.**

You should complete each exercise and question; your project will be evaluated against [this rubric](https://review.udacity.com/#!/rubrics/2516/view).

## Project Evaluation

You will be graded on your implementation of a plagiarism detector as well as complete answers to any questions in the project notebook. You'll submit a **zip file** or Github repo that includes complete notebooks, with all cells executed, and you'll be graded according to the project rubric.

## Exploring the Data

Before starting the project, you are given the option to explore the plagiarism data you'll be working with, in the **next workspace**.